AMENDMENTS TO THE CLAIMS

The following listing of claims replaces all prior versions of the claims and all prior listings of the claims in the present application.

1. (previously presented) An incrementer, comprising:

a 4-bit zero detection unit, wherein said 4-bit zero detection unit groups every 4 bits of an operand, starting from the least significant bit (LSB), determines whether each 4-bit group includes a first logic state, and outputs a second logic state as first logic state inclusion information for each 4-bit group if the 4-bit group includes the first logic state and outputs the first logic state as the first logic state inclusion information for each 4-bit group if the 4-bit group does not include the first logic state;

a flag information generation unit, wherein said flag information generation unit outputs flag information for each 4-bit group by generating the first logic state for a first group with the second logic state, starting from the LSB of the first logic state inclusion information for each 4-bit group, and for following lower order groups, and by generating the second logic state for higher order groups preceding the first group with the second logic state;

a 4-bit increment unit, wherein said 4-bit increment unit receives the operand and performs an increment on each 4-bit group; and

an increment output unit, wherein said increment output unit performs a logical combination on the operand, the first logic state inclusion information for each 4-bit group, the flag information for each 4-bit group, and an increment value for each 4-bit group, and generates a whole increment value by outputting 4 bits of the first logic state for each 4-bit group, 4 bits of the operand for each 4-bit group, or 4 bits of the increment value for each 4-bit group.

2. (previously presented) The incrementer of claim 1, wherein the increment is performed in accordance with the following Boolean logic expressions,

IF IN<0> and IN<1> = "1",

(IN+1)<3:2> = IN_C<3:2>,

(IN+1)<1:0> = "00",

IF IN<0> or IN<1> = "0",

(IN+1)<3:2> = IN<3:2>,

(IN+1)<1:0> = IN_C<1:0>,

where IN represents the operand, IN+1 represents the increment value, IN_C represents a newly defined increment value, "0" represents the first logic state, and "1" represents the second logic state, and

IN_C<0> = ~IN<0>,

IN_C<1> = IN<0> OR IN<1>,

IN_C<2> = ~IN<2>,

IN_C<3> = IN<2> OR IN<3>,

where IN represents the operand and IN_C represents the newly defined increment value.

3. (previously presented) The incrementer of claim 1, wherein the logical combination is performed in accordance with the following Boolean expression,

IF ZD = "0" and CA = "0", IO = "0000",

IF ZD = "1" and CA = "0", IO = IN+1,

IF CA = "1" (ZD = "0" or "1"), IO = IN,

where ZD represents the first logic state inclusion information for each 4-bit group, CA represents the flag information for each 4-bit group, IO represents the whole increment value for each 4-bit group, IN represents the operand, IN+1 represents the increment value, “0” represents the first logic state, and “1” represents the second logic state.

4. (previously presented) The incrementer of claim 1, wherein the incrementer operates when the 4-bit zero detection unit, the flag information generation unit, the 4-bit increment unit, and the increment output unit are activated in response to a clock signal.

5. (previously presented) The incrementer of claim 4, wherein when the clock signal is inactivated, respective input nodes of output buffers of the 4-bit zero detection unit, the flag information generation unit, the 4-bit increment unit, and the increment output unit are precharged to a precharging voltage.

6. (original) The incrementer of claim 5, wherein an inverter for inverting the precharging voltage and a PMOSFET for supplying the precharging voltage to an input end of the inverter in response to an output of the inverter are used as the output buffers.

7. (previously presented) The incrementer of claim 6, wherein the output of the inverter is inverted to the second logic state when the clock signal is activated and a plurality of NMOSFETs, connected in series between the output buffers and a ground voltage, are activated.

8. (currently amended) A[[n]] microprocessor-implemented increment method of an incrementer, the method comprising:

grouping every 4 bits of an operand, starting from the least significant bit (LSB);

determining whether a first logic state is included in each 4-bit group;

outputting first logic state inclusion information for each 4-bit group as a second logic state if the first logic state is included and as the first logic state if the first logic state is not included;

outputting flag information for each 4-bit group by generating the first logic state for a first group with the second logic state, starting from the LSB of the first logic state inclusion information for each 4-bit group, and for following lower order groups, and by generating the second logic state for higher order groups preceding the first group with the second logic state;

incrementing, within the microprocessor, each 4-bit group;

performing a logical combination on the operand, the first logic state inclusion information for each 4-bit group, the flag information for each 4-bit group, and an increment value for each 4-bit group; [[and]]

generating a whole increment value by outputting 4 bits of the first logic state for each 4-bit group, 4 bits of the operand for each 4-bit group, or 4 bits of the increment value for each 4-bit group; and

outputting the whole increment value ~~wherein the method is implemented in a microprocessor~~.

9. (previously presented) The method of claim 8, wherein incrementing each 4-bit group is performed in accordance with the following Boolean logic expressions,

IF IN<0> and IN<1> = "1",

(IN+1)<3:2> = IN_C<3:2>,

(IN+1)<1:0> = "00",

IF IN<0> or IN<1> = "0",

(IN+1)<3:2> = IN<3:2>,

(IN+1)<1:0> = IN_C<1:0>,

where IN represents the operand, IN+1 represents the increment value, IN_C represents a newly defined increment value, "0" represents the first logic state, and "1" represents the second logic state, and

IN_C<0> = ~IN<0>,

IN_C<1> = IN<0> OR IN<1>,

IN_C<2> = ~IN<2>,

IN_C<3> = IN<2> OR IN<3>,

where IN represents the operand and IN_C represents the newly defined increment value.

10. (previously presented) The method of claim 8, wherein the logical combination is performed in accordance with the following Boolean expression,

IF ZD = "0" and CA = "0", IO = "0000",

IF ZD = "1" and CA = "0", IO = IN+1,

IF CA = "1" (ZD = "0" or "1"), IO = IN,

where ZD represents the first logic state inclusion information for each 4-bit group, CA represents the flag information for each 4-bit group, IO represents the whole increment value for each 4-bit group, IN represents the operand, IN+1 represents the increment value, "0" represents the first logic state, and "1" represents the second logic state.

11. (previously presented) The method of claim 8, wherein the incrementer operates when a 4-bit zero detection unit, a flag information generation unit, a 4-bit increment unit, and an increment output unit are activated in response to a clock signal.

12. (previously presented) The method of claim 11, wherein when the clock signal is inactivated, respective input nodes of output buffers of the 4-bit zero detection unit, the flag information generation unit, the 4-bit increment unit, and the increment output unit are precharged to a precharging voltage.

13. (previously presented) The method of claim 12, wherein an inverter for inverting the precharging voltage and a PMOSFET for supplying the precharging voltage to an input end of the inverter in response to the output of the inverter are used as output buffers.

14. (previously presented) The method of claim 13, wherein the output of the inverter is inverted to the second logic state when the clock signal is activated and a plurality of NMOSFETs, connected in series between the output buffers and a ground voltage, are activated.

15. (previously presented) An incrementer, comprising:

a b-bit zero detection unit, wherein b is a number of bits greater than 3, wherein the b bits of an operand are grouped in an order resulting in b-bit groups, and wherein said b-bit detection unit outputs first logic state inclusion information;

a flag information generation unit, wherein said flag information generation unit outputs flag information for each b-bit group;

a b-bit increment unit, wherein said b-bit increment unit receives the operand and performs an increment on each b-bit group; and

an increment output unit, wherein said increment output unit performs a logical combination and generates a whole increment value by outputting b bits of a first logic state for each b-bit group, b bits of the operand for each b-bit group, or b bits of an increment value for each b-bit group.

16. (previously presented) The incrementer of claim 15, wherein b is 4.

17. (previously presented) The incrementer of claim 15, wherein the b bits of the operand are grouped in an order string, starting from the least significant bit (LSB).

18. (previously presented) The incrementer of claim 17, wherein the b-bit zero detection unit determine whether each b-bit group includes the first logic state, and outputs a second logic state as the first logic state inclusion information for each b-bit group if the b-bit group includes the first logic state and outputs the first logic state as the first logic state inclusion information for each b-bit group if the b-bit group does not include the first logic state.

19. (previously presented) The incrementer of claim 18, wherein the flag information generation unit outputs flag information by generating the first logic state for a first group with the second logic state, starting from the least significant bit (LSB) of the first logic state inclusion information for each b-bit group, and for following lower order groups, and by generating the second logic state for higher order groups preceding the first group with the second logic state.

20. (previously presented) The incrementer of claim 19, wherein the logical combination operates on the operand, the first logic state inclusion information for each b-bit group, the flag information for each b-bit group, and the increment value for each b-bit group.

21. (currently amended) A[[n]] microprocessor-implemented increment method, comprising:

grouping every b-bits of an operand to form b-bit groups;

determining whether a first logic state is included in each b-bit group;

outputting first logic state inclusion information for each b-bit group;

outputting flag information for each b-bit group;

incrementing each b-bit group by an increment value;

performing, within the microprocessor, a logical combination on the operand, the first logic state inclusion information for each b-bit group, the flag information for each b-bit group, and the increment value for each b-bit group; [[and]]

generating a whole increment value; and

outputting the whole increment value ~~wherein the method is implemented in a microprocessor~~.

22. (original) The method of claim 21, wherein b is 4.

23. (previously presented) The method of claim 21, wherein grouping starts from the least significant bit (LSB) of the operand.

24. (previously presented) The method of claim 23, wherein outputting a first logical state inclusion information comprises:

outputting a second logic state if the first logic state is included in the b-bit group; and

outputting the first logic state if the first logic state is not included in the b-bit group.

25. (previously presented) The method of claim 24, wherein the step of outputting flag information comprises:

generating the first logic state for a first group with the second logic state, starting from the least significant bit (LSB) of the first logic state inclusion information for each b-bit group, and for following lower order groups; and

generating the second logic state for higher order groups preceding the first group with the second logic state.

26. (previously presented) The method of claim 25, wherein the whole increment value is generated by outputting b bits of the first logic state for each b-bit group, b bits of the operand for each b-bit group, or b bits of the increment value for each b-bit group.

27. (previously presented) An incrementer, comprising:

a b-bit zero detection means, wherein b is a number of bits greater than 3, wherein said b-bit zero detection means groups every b bits of an operand, starting from the least significant bit (LSB), determines whether each b-bit group includes a first logic state, and outputs a second logic state as first logic state inclusion information for each b-bit group if the b-bit group includes the first logic state and outputs the first logic state as the first logic state inclusion information for each b-bit group if the b-bit group does not include the first logic state;

a flag information generation means, wherein said flag information generation means outputs flag information for each b-bit group by generating the first logic state for a first group with the second logic state, starting from the LSB of the first logic state inclusion information for each b-bit group, and for following lower order groups, and by generating the second logic state for higher order groups preceding the first group with the second logic state;

a b-bit increment means, wherein said b-bit increment means receives the operand and performs an increment on each b-bit group; and

an increment output means, wherein said increment output means performs a logical combination on the operand, the first logic state inclusion information for each b-bit group, the flag information for each b-bit group, and an increment value for each b-bit group, and generates a whole increment value by outputting b bits of the first logic state for each b-bit group, b bits of the operand for each b-bit group, or b bits of the increment value for each b-bit group.

28. (previously presented) The incrementer of claim 27, wherein b is 4.